第4次实验

1. 字符串处理

考虑char 数组表示的字符串，比如str[]=” Abc 789 ”，现在要去掉该字符串头部、尾部的空格，中间的空格保留。

## 1 去掉头部尾部空格

【分析】

删除char数组头部和尾部的空格，首先需知道数组中从前到后的第一个非空字符，和最后一个非空字符。

【算法】

// 去除左侧空格

// strIn——输入串，结果放在输入串自身

void trimLeft(char \*strIn)

{

从前向后循环找到strln中第一个非空格位置，记作i位置

// 将strln中从i到末尾copy到头部，这一步也可以用循环实现

strncpy(strIn, strIn + i, strlen(strIn) - i);

}

// 去除右侧空格

// strIn——输入串，结果放在输入串自身

void trimRight(char \*strIn)

{

从后向前循环找到strln中最后一个非空格位置，记作j位置

修改j + 1位置为 '\0';

}

## 2 将字符型数字变为真实的数字

考虑char 数组表示的正整数数字，比如”123”、”976”，现在要将他们变为int型数字，然后将两个数字作加法。

【分析】

一位的字符型数字减去字符’0‘即为该整数，如 ’1‘-’0‘ = 1、’7‘-’0‘ = 7。

【算法】

int convertToInt(char \*p)

{

设sum = 0;

循环（i从0到字符末尾）

{

sum = 10 \* sum + p[i] - '0';

}

return sum;

}

1. 整数处理

## 判断正整数是否左右对称

所谓对称数，就是譬如：1，323，34543之类的数。可以把原来的数的高低位互换，构造一个新的数，如果和原来的数相等则是对称数。

【编程参考】

操作1：若tmp为正整数，则tmp %10计算其个位数

操作2：而tmp = tmp/10 则每次将tmp去掉各位数字

将操作1、2放入循环，则可依次求得每一位的数字。

若一个数字个位、十位、百位、……，依次为P0、P1、P2、…等，则

循环执行：

S = S\*10 + Pi  // S 初值为 0

可以计算出原整数tmp倒转以后的数字